Lens Aberrations Detection and Digital Camera Identification with Convolutional Autoencoders

Jarosław Bernacki[1][a] and Rafał Scherer[2][b]
[1]*Department of Artificial Intelligence, Częstochowa University of Technology, al. Armii Krajowej 36, 42-200 Częstochowa, Poland*
[2]*Faculty of Computer Science, AGH University of Krakow, Poland*

Keywords: Privacy, Security, Convolutional Neural Networks, Convolutional Autoencoders, Digital Forensics, Digital Camera Identification, Hardwaremetry.

Abstract: Digital camera forensics relies on the ability to identify digital cameras based on their unique characteristics. While many methods exist for camera fingerprinting, they often struggle with efficiency and scalability due to the large image sizes produced by modern devices. In this paper, we propose a novel approach that utilizes convolutional and variational autoencoders to detect optical aberrations, such as vignetting and distortion. Our model, trained in an aberration-independent manner, enables automatic detection of these distortions without needing reference patterns. Furthermore, we demonstrate that the same methodology can be applied to digital camera identification based on image analysis. Extensive experiments conducted on multiple cameras and images confirm the effectiveness of our approach in both aberration detection and device fingerprinting, highlighting its potential applications in forensic investigations.

1 INTRODUCTION

Digital forensics is a field that has garnered significant attention in recent years. One of the most prominent topics in digital forensics is the identification of imaging sensors in digital cameras. Digital cameras have become widely accessible and affordable, contributing to their popularity. Even more prevalent are smartphones and mobile devices, commonly equipped with built-in digital cameras. This widespread availability encourages people to take photos and share them on social media networks. However, the capability to determine whether an image was taken by a specific camera can pose a serious threat to user privacy. Consequently, a substantial body of research in recent years has focused on studying imaging device artifacts that can be used for digital camera identification.

Digital camera identification can be approached in two primary ways: individual source camera identification (ISCI) and source model camera identification (SCMI). ISCI can distinguish a specific camera among cameras of the same model and different models. In contrast, SCMI can only differentiate between different camera models but not between individual cameras of the same model. For example, given cameras such as Canon EOS R (0), Canon EOS R (1), ..., Canon EOS R (n), Nikon D780 (0), Nikon D780 (1), Sony A1 (0), and Sony A1 (1), ISCI would differentiate each camera, while SCMI would only identify the general models (Canon EOS R, Nikon D780, Sony A1). This limitation of SCMI motivates the development of methods and algorithms focused on the ISCI aspect.

The most common methods for digital camera identification are based on Photo-Response Non-Uniformity (PRNU). These methods compare the noise patterns of a given image with the known noise pattern of a camera. PRNU arises from imperfections in the image sensor, creating a unique pattern for each camera. This pattern can be estimated from multiple images captured by a camera and used as a reference for identification. If the PRNU patterns match, the image was likely captured by that camera. PRNU-based methods are widely used due to their robustness against post-capture processing and compression.

A state-of-the-art algorithm for ISCI was proposed by Luk'as et al. (Lukás et al., 2006), utilizing PRNU for camera identification. The PRNU $\mathbf{K}$ may be calculated in the following manner: $\mathbf{K} = \mathbf{I} - F(\mathbf{I})$,

[a] https://orcid.org/0000-0002-4488-3488
[b] https://orcid.org/0000-0001-9592-262X

Bernacki, J. and Scherer, R.
Lens Aberrations Detection and Digital Camera Identification with Convolutional Autoencoders.
DOI: 10.5220/0013485900003979
In *Proceedings of the 22nd International Conference on Security and Cryptography (SECRYPT 2025)*, pages 84-95
ISBN: 978-989-758-760-3; ISSN: 2184-7711

where $\mathbf{I}$ is the input image and F is a denoising filter. This PRNU acts as a unique fingerprint for the camera. Many studies have confirmed the high efficacy of this method. However, a significant drawback is that the camera's fingerprint is represented as a matrix with the original image dimensions, posing storage challenges for large numbers of PRNUs in forensic centers. This issue drives the need for methods to minimize this storage problem.

Another approach involves feature-based methods, which extract features such as metadata, color balance, geometric distortion, lens artifacts, etc., and match them with known camera features. The most recent family of methods utilizes deep learning, typically employing convolutional neural networks (CNNs) to extract features from images and compare them with features from known cameras (Bondi et al., 2017; Ding et al., 2019; Kirchner and Johnson, 2020; Li et al., 2018; Lukás et al., 2006; Mandelli et al., 2020; Yao et al., 2018). Additionally, various hybrid methods combine multiple algorithms to enhance the accuracy of camera identification.

Lens aberration identification is a critical aspect of digital forensics. Despite technological advancements, both digital single-lens reflex (DSLR) and mirrorless cameras continue to suffer from various optical issues. Common deviations include chromatic aberrations, dispersion, vignetting, distortion, and coma. Vignetting, which can result from optical defects or sensor imperfections (Lopez-Fuentes et al., 2015; Ray, 2002), is a reduction of the image brightness near the edges, leading to darker corners. This flaw is especially prevalent in compact and DSLR cameras. Types of vignetting are thoroughly detailed in (Lopez-Fuentes et al., 2015), and the problem has garnered significant research attention, with numerous algorithms (De Silva et al., 2016; Kordecki et al., 2017) and patents (Lee et al., 2017) proposed for its correction. Another related issue is lens distortion, which occurs as a deviation from rectilinear projection (Park and Hong, 2001; Claus and Fitzgibbon, 2005). This distortion causes straight lines to appear curved in images and is characterized by changes in magnification relative to the image's distance from the optical axis (Goljan and Fridrich, 2014).

1.1 Contribution

The contribution of this paper is twofold. Firstly, we propose a method that utilizes a convolutional (CAE) and variational (VAE) autoencoder to identify images that show different types of lens aberrations, including lens vignetting and distortion. We experimentally show that our methods are capable of detecting lens vignetting based on several models of lenses, as well as detecting lens distortion with a reliable probability. Secondly, we show that the proposed autoencoders may be used to identify digital cameras in the ISCI aspect based on images. We demonstrate that this approach requires less time for training, which may speed up the image processing workflow. Our experiments, conducted on a large set of modern digital cameras, confirm that the accuracy of our method is comparable to state-of-the-art methods. Additionally, we perform a statistical analysis of the obtained results, which further confirms their reliability.

1.2 Organization of the Paper

The paper is organized as follows. The next section discusses previous and related works. In Section 3 the problem formulation and the proposed method are described. Section 4 presents the results of classification compared with the state-of-the-art methods. In Section 5 there are described the results of statistical analysis. The final section concludes this work.

2 PREVIOUS AND RELATED WORK

In (Baar et al., 2012), the authors proposed utilizing the k-means algorithm to manage photo response non-uniformity (PRNU) patterns. These patterns are compared using correlation and then grouped by the k-means algorithm. As a result, patterns grouped within the same cluster are considered to belong to the same camera. Experiments conducted on a database of 500 images showed that images within a cluster had a true positive rate (TPR) of 98% for belonging to a particular camera. In (Julliand et al., 2016), the authors demonstrate that different types of noise significantly affect raw images. They show that JPEG lossy compression generates noise that impacts groups of pixels. A specific example highlighted how an image's histogram changes before and after saving it as a JPEG, indicating that JPEG compression introduces distinct artifacts that can be used for identification purposes. In (Taspinar et al., 2016), the feasibility of sensor recognition for image blocks smaller than 50×50 pixels is explored. The study uses the peak-to-correlation energy (PCE) ratio for verification, and results indicate that analyzing such small blocks with low PCE values is inefficient. The objective of (Jiang et al., 2016) is to determine if images across several social network accounts were taken by the same user. The authors employ the formula from (Lukás et al., 2006) to find the camera fingerprint and cluster im-

ages based on correlation. Experiments with 1576 images evaluated performance using precision and recall measures, achieving a clustering precision of 85% and a recall of 42%.

The analysis of how image features affect PRNU is discussed in (Tiwari and Gupta, 2018). Intensity-based features and high-frequency details, such as edges and textures, impact the final quality of the camera's fingerprint. To enhance this quality, a weighting function (WF) is proposed. Initially, regions of the image that provide reliable and unreliable PRNU are estimated. Then, the WF assigns higher weights to regions yielding reliable PRNU and lower weights to those producing less reliable PRNU. In (Marra et al., 2018), the vulnerability of deep learning approaches to adversarial attacks in digital camera identification is examined. The goal is to demonstrate how to deceive a CNN-based classifier to produce incorrect camera identification. The study describes several scenarios where the image undergoes lossless or lossy compression. Attacks on the classifier are performed using methods such as the Fast Gradient Sign Method (FGSM)(Goodfellow et al., 2015), DeepFool(Moosavi-Dezfooli et al., 2016), and the Jacobian-based Saliency Map Attack (JSMA) (Papernot et al., 2016). FGSM involves adding additive noise to the image, which can sometimes visibly affect image quality. To mitigate this, DeepFool uses local linearization of the classifier under attack. JSMA, on the other hand, is a greedy iterative procedure that computes a saliency map at each iteration, identifying the pixels that most influence correct classification. Experiments demonstrated that these attack methods can effectively deceive CNN-based classifiers.

3 PROPOSED AUTOENCODERS

3.1 Preliminaries and Problem Formulation

Camera's Fingerprint. Let M be the number of images from camera $A^{(i)}$. In order to learn the specificity of the camera (but not the content of the input image), we denoise the cameras' images. We use the well-known formula presented as eq. 1, utilized in (Lukás et al., 2006; Tuama et al., 2016) to calculate the residuum $\mathbf{K}_j$ for the j-th image of camera $A^{(i)}$:

$$\mathbf{K}_j = \mathbf{I}_j - F(\mathbf{I}_j) \tag{1}$$

where $\mathbf{I}_j$ is the j-th image of camera $A^{(i)}$, and F stands for a denoising filter. To obtain the fingerprint $\mathbf{K}^{(\mathbf{i})}$ of the camera $A^{(i)}$ we calculate:

$$\mathbf{K}^{(\mathbf{i})} = \frac{1}{M}\sum_{k=1}^{M}\mathbf{K}_k^{(i)} \tag{2}$$

According to (Lukás et al., 2006), the procedure described as eq. 2 is representative, if $M > 45$.

Camera Identification. Let us define the camera identification task as the statistical approach.

Definition 1. (Camera identification task) Let N be the number of cameras. For the image $\mathbf{I}$ define, from which camera $A^{(i)}$ (where $i \in \{1,2,\ldots,N\}$) this image comes from. We define the following hypotheses:

- The null hypothesis ($\mathcal{H}_0$): The image $\mathbf{I}$ comes from the camera $A^{(i)}$;
- The alternative hypothesis ($\mathcal{H}_1$): The image $\mathbf{I}$ does not come from the camera $A^{(i)}$.

To verify the hypotheses we have to define the statistical test $T(\mathbf{I})$, which measures the compatibility of the fingerprint $\mathbf{K}^{(\mathbf{i})}$ from the camera $A^{(i)}$ with a new residuum $\mathbf{K}_x$ (eq. 3):

$$T(\mathbf{I}) = D(\mathbf{K}^{(\mathbf{i})}, \mathbf{K}_x) \tag{3}$$

where D may be a distance function or correlation coefficient. We reject the null hypothesis $\mathcal{H}_0$, whether:

$$T(\mathbf{I}) > \tau \tag{4}$$

where τ is a rejection threshold (the critical value) on the significance level α.

The image $\mathbf{I}$ is considered as made with the camera $A^{(i)}$, if the test statistic meets the criterion:

$$T(\mathbf{I}) \leq \tau \tag{5}$$

If for all cameras $T(\mathbf{I}) > \tau$, then the image $\mathbf{I}$ was not made by any of the considered cameras.

Aberrations. In this paper, we consider the following lens aberrations: vignetting and distortion. Lens vignetting is a defect that manifests itself as a decrease in the brightness of an image, usually in all its corners relative to the center (Lanh et al., 2007; Kordecki et al., 2015). Formally, lens vignetting can be modeled as presented in Def. 2.

Definition 2. (Lens vignetting) Let $I_0(x_0, y_0)$ denote the brightness in the middle of the image $\mathbf{I}$. The vignetting may be defined as the following:

$$I(r_{(v)}) = I_0 \cdot \left(1 - k \cdot \left(\frac{r_{(v)}}{r_{(v)}^{\max}}\right)^2\right) \tag{6}$$

where:

- $I(r_{(v)})$ – brightness in a selected pixel with $r_{(v)}$ distance from the middle of the image $\mathbf{I}$;

- $I_0(x_0, y_0)$ – brightness in the middle of the image $\mathbf{I}$;
- k – vignetting coefficient;
- $r_{(v)}^{\max}$ – maximum distance from the middle of the image $\mathbf{I}$.

The $r_{(v)}$ may be calculated using the Euclidean distance $r_{(v)} = \sqrt{(x-x_0)^2 + (y-y_0)^2}$ for any pixel (x, y).

Lens distortion is an optical defect that is visible in photos as a distortion of shapes due to deviation from a rectilinear projection (e.g. the edge of a building in a distorted photo appears to be deviated from the vertical, etc.). The most common types of distortion are barrel distortion and pincushion distortion. In barrel distortion, the center of the frame is emphasized, while in pincushion distortion, the center of the frame looks "sunken" (Goljan and Fridrich, 2014). Simplifying a bit, distortion occurs when lines created by pixels, which in the real world should be vertical or horizontal, are not parallel to its edges or are curved in the photo. Formally, lens distortion can be described as in Def. 3.

Definition 3. (Lens distortion) For an image without distortion, the pixel coordinates (x, y) are mapped to the coordinates (x', y') in the distorted image according to the equation:

$$r'_{(d)} = r_{(d)} \cdot (1 + k_1 r_{(d)}^2 + k_2 r_{(d)}^4) \tag{7}$$

where:

- $r_{(d)}$ is the distance of the pixel from the center of the image;
- k_1, k_2 – distortion coefficients;
- $r'_{(d)}$ – the distance of the distorted pixel from the center.

Let us define the task of aberration detection, which is considered as lens vignetting and lens distortion detection, using the convolutional autoencoder.

Definition 4. (Aberration detection) Let $\mathcal{D} = \{\mathbf{I}_j, \mathbf{Y}_j\}_{j=1}^{M}$ denote the training set. The $\mathbf{I}_j$ is an input image, and $\mathbf{Y}_j$ denotes the mask corresponding to the image $\mathbf{I}_j$, where each value $y_{jk} \in \{0, 1\}$ represents every pixel of $\mathbf{I}_j$ if it shows the aberration (1) or not (0). The $\mathbf{I}_j$ has dimensions $H \times W \times C$, where H is the image height, W stands for the image width and C is the number of color channels (typically for RGB images $C = 3$).

Let f_θ be the convolutional autoencoder with parameters θ. The autoencoder f_0 transforms the image $\mathbf{I}_j$ into the matrix of predictions $\hat{\mathbf{Y}}_j$, where $\hat{\mathbf{Y}}_j = f_\theta(\mathbf{I}_j)$. The dimensions of $\hat{\mathbf{Y}}_j$ are $H \times W$, where $\hat{y}_{jk} \in [0, 1]$ is the probability that the k-th pixel shows the aberration.

Definition 5. (Loss function) For the image $\mathbf{I}_j$ the binary cross-entropy function is defined as:

$$\Lambda(\mathbf{I}_j, \mathbf{Y}_j, \hat{\mathbf{Y}}_j) = -\frac{1}{HW} \sum_{k=1}^{H \times W} [y_{jk} \log(\hat{y}_{jk}) + (1 - y_{jk}) \log(1 - \hat{y}_{jk})] \tag{8}$$

where:

- y_{jk} is the label for the pixel k;
- $\hat{y}_{jk}$ is the probability for the pixel k;
- H and W are dimensions of the image $\mathbf{I}_j$.

Definition 6. (Minimizing the loss function) During the training the parameters θ are optimized, minimizing the loss function:

$$\min_\theta \frac{1}{M} \sum_{j=1}^{M} \Lambda(\mathbf{I}_j, \mathbf{Y}_j, \hat{\mathbf{Y}}_j) \tag{9}$$

As the optimization algorithm, Adam (Kingma and Ba, 2015) is used.

Definition 7. (Aberration prediction) The autoencoder f_θ generates for the new image $\mathbf{I}_x$ the matrix of predictions $\hat{\mathbf{Y}}_x = f_\theta(\mathbf{I}_x)$. To receive the final decision on aberration, the thresholding is used in the following manner:

$$\hat{y}_{x,k} = \begin{cases} 1 & \text{if } \hat{y}_{x,k} \geq \gamma \\ 0 & \text{if } \hat{y}_{x,k} < \gamma \end{cases}$$

where γ is a threshold, for instance $\gamma = 0.5$ decides if the pixel k is marked as aberrated (1) or not (0).

3.2 The Proposed Autoencoders

Convolutional Autoencoder. The convolutional autoencoder (CAE) is a classic autoencoder approach that consists of two main parts: the encoder (encoding part), and the decoder (the decoding part). The encoder reduces the input resolution gradually as the number of channels increases. The decoder, conversely, restores the original resolution. We propose the following structure of the CAE:
The encoding part:

(1) A first convolutional layer of 32 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(2) A second convolutional layer of 64 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(3) A third convolutional layer of 128 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(4) A fourth convolutional layer of 256 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1.

The decoding part:

(1) A first transposed convolutional layer of 128 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(2) A second transposed convolutional layer of 64 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(3) A third transposed convolutional layer of 32 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(4) A fourth transposed convolutional layer with a filter of size 3×3 (stride 2), with sigmoid as an activation function, followed by a Max-Pooling layer + padding 1.

The encoder consists of four convolutional layers, which gradually reduce the image resolution and increase the number of channels. Each convolutional layer acts as a filter, capturing increasingly abstract features – from edges and textures to more complex structures. As we go through the layers, information about details is lost, but important features are preserved, representing the image in a compressed way. After the last layer, we get a low-dimensional representation (so-called latent vector).

The decoder reverses the encoder's operation – it uses transposed convolutional layers (deconvolution) to restore the original image size. Each step gradually increases the resolution, reconstructing missing details. The final layer returns an image in the range of values [0,1], usually using a sigmoid activation function. The model learns to minimize the difference between the input and the output, so it can effectively eliminate noise or detect anomalies when the reconstruction does not match the input (e.g. distortion or vignetting).

Variational Autoencoder. The structure of the variational autoencoder (VAE) is generally similar to the structure of CAE, but additionally, the VAE uses linear layers (fully connected) to encode an image into the latent vector. Let us discuss the proposed VAE with the following structure:

The encoding part:

(1) A first convolutional layer of 32 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(2) A second convolutional layer of 64 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(3) A third convolutional layer of 128 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(4) A fourth convolutional layer of 256 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(5) Fully connected layers which generate the μ (mean) and **logvar** (log variance) parameters to model the latent vector.

The decoding part:

(1) A first transposed convolutional layer of 128 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(2) A second transposed convolutional layer of 64 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(3) A third transposed convolutional layer of 32 filters of size 3×3 (stride 2), with ReLU as an activation function, followed by a Max-Pooling layer + padding 1;

(4) A fourth transposed convolutional layer with a filter of size 3×3 (stride 2), with sigmoid as an activation function, followed by a Max-Pooling layer + padding 1.

Similar to a classic autoencoder, the encoder consists of convolutional layers that reduce the image resolution and extract key features. However, instead of returning a latent vector, the model generates two vectors – μ (mean) and **logvar** (log variance), which define a normal distribution in the latent vector. Instead of direct encoding, the model samples values from this distribution (using reparameterization), which introduces an element of randomness and allows the generation of new images.

The decoder works similarly to CAE, but instead of reconstructing the image from a latent vector, it does so from a sample taken from a Gaussian distribution. This allows the model to generate diverse images, even if the input image is the same. The decoder gradually increases the resolution through transposed convolutional layers and ends with a layer with sigmoid activation, returning the image reconstruction.

The Discriminator. In order to classify images, the discriminator may be used. The idea of the discriminator is based on the Generative Adversarial Network (GAN) (Goodfellow et al., 2014). The use of the discriminator is essential to classify the images produced by autoencoder decoders. We propose to use as the discriminator a convolutional neural network, however, well-known machine learning algorithms, such as Support Vector Machine (SVM) might also be used.
The structure of the sample discriminator is described below:

(1) A first convolutional layer of 32 filters of size 3×3 with ReLU as an activation function, stride 2, followed by a max-pooling layer;

(2) A second convolutional layer of 64 filters of size 3×3 with ReLU as an activation function, stride 2, followed by a max-pooling layer;

(3) A third convolutional layer of 128 filters of size 3×3 with ReLU as an activation function, stride 2, followed by a max-pooling layer;

(4) Fully connected 512 + dropout 0.5 + ReLU;

(5) Fully connected 128 + dropout 0.5.

The activation function is softmax.

All meta-parameters both for the proposed autoencoders, as well the discriminator were determined experimentally.

4 EXPERIMENTAL EVALUATION

We conduct two experiments. The first experiment presents the results of lens aberrations detection, including vignetting and distortion identification with the proposed autoencoders. The second experiment shows the standard identification procedure for both of the proposed CAE and VAE, as well as considered state-of-the-art methods.

4.1 Experimental Setup

Datasets. For both experiments, we use images coming from the IMAGINE dataset (Bernacki. and Scherer., 2023).

For experiment I, we use both images from the IMAGINE dataset but also blank images that may make learning the lens vignetting and lens distortion. Firstly, the autoencoders are trained with aberration-free images. To determine whether the images have distorted pixels, we have used the Hugin Photo Stitcher software (hug, enet). For each case (no vignetting; not distorted) at least 30 images were used (per lens). Sample images used for training are shown as Fig. 1 and 2. The following cameras and lenses were used for the Experiment I:

Nikon D3100:

1. Nikon Nikkor AF-S DX 18-105 mm f/3.5-5.6 VR ED
2. Nikon Nikkor AF-S DX 35 mm f/1.8G

Nikon D7200:

1. Nikon Nikkor AF-P DX 10-20 mm f/4.5-5.6G VR
2. Nikon Nikkor AF-S DX 18-55 mm f/3.5-5.6G VR ED
3. Nikon Nikkor AF-S DX Micro 40 mm f/2.8G

Nikon D750:

1. Nikon Nikkor AF-S 20 mm f/1.8G ED
2. Nikon Nikkor AF 50 mm f/1.8D

Panasonic GX80:

1. Panasonic G VARIO 14-42 mm f/3.5-5.6 MEGA O.I.S
2. Olympus M.Zuiko Digital ED 30 mm f/3.5 Macro

For experiment II, we use a set of 17 modern cameras that include Canon EOS 1D X Mark II (C1), Canon EOS 5D Mark IV (C2), Canon EOS M5 (C3), Canon EOS M50 (C4), Canon EOS R (C5), Canon EOS R6 (C6), Canon EOS RP (C7), Fujifilm X-T200 (F1), Nikon D5 (N1), Nikon D6 (N2), Nikon D500 (N3), Nikon D780 (N4), Nikon D850 (N5), Nikon Z6 II (N6), Nikon Z7 II (N7), Sony A1 (S1), Sony A9 (S2). At least 30 images per camera are used for learning.

Evaluation Measures. As evaluation, we use standard *accuracy* (ACC), defined as:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

where TP/TN denotes "true positive/true negative"; FP/FN stands for "false positive/false negative". TP denotes the number of cases correctly classified to a specific class; TN refers to instances that are correctly rejected. FP denotes cases incorrectly classified to the specific class; FN is cases incorrectly rejected.

Implementation. Experiments are held on a Gigabyte Aero notebook equipped with an Intel Core i7-13700H CPU with 32 gigabytes of RAM and a Nvidia GeForce RTX 4070 GPU with 8 gigabytes of video memory. Scripts for CNNs are implemented in Python under the PyTorch framework (with Nvidia CUDA support).

(a) (b) (c)

Figure 1: Sample images for lens vignetting identification: (a) – blank vignetted image; (b) and (c) – blank and natural (respectively) non-vignetted images (Nikon D750 + Nikon Nikkor AF-S 20 mm f/1.8G ED).

(a) (b)

Figure 2: Sample images for lens distortion identification: (a) – distorted image; (b) – non-distorted image (Nikon D3100 + Nikon Nikkor AF-S DX 18-105 mm f/3.5-5.6 VR ED).

4.2 State-of-the-Art Methods – Recall

Let us recall some methods for a digital camera identification.

Mandelli et al.'s CNN. Let us briefly recall the structure of Mandelli et al.'s (Mandelli et al., 2020) convolutional neural network (CNN):

(1) A first convolutional layer of kernel 3×3 producing feature maps of size 16×16 pixels with Leaky ReLU as an activation method and max-pooling;

(2) A second convolutional layer of kernel 5×5 producing feature maps of size 64×64 pixels with Leaky ReLU as an activation method and max-pooling;

(3) A third convolutional layer of kernel 5×5 producing feature maps of size 64×64 pixels with Leaky ReLU as an activation method and max-pooling;

(4) A pairwise correlation pooling layer;

(5) Fully connected layers (FC).

For more details related to the structure of the network, we refer to the authors' paper.

Kirchner & Johnson's CNN. Kirchner and Johnson (Kirchner and Johnson, 2020) proposed a following network:

1. 17 layers implementing 64 convolutional filters with 3×3 kernels;
2. ReLU as an activation method after each layer;
3. Fully connected layers (FC).

All aforementioned convolutional neural networks are trained by noise residuals calculated with the denoising formula (Eq. 10).

Lukás et al.'s Algorithm. The non-convolutional Lukás et al.'s algorithm (Lukás et al., 2006) is based on the calculation of the noise residual $\mathbf{K}$ as shown in Equation 10.

$$\mathbf{K} = \mathbf{I} - F(\mathbf{I}), \tag{10}$$

where F is a denoising (usually wavelet-based) filter, the $\mathbf{K}$ stands for a single noise residual of one image $\mathbf{I}$. To obtain a representative noise residual of the camera, this procedure should be repeated for at least 45 images. The camera's noise residual is finally calculated as an average of a particular number of single noise residuals. It is recommended to process images in their original resolution.

4.3 Experiment I – Detecting Lens Aberrations

In this experiment, we tested our proposed autoencoders to see if they were capable of detecting images with lens aberrations that were considered vignetting and distortion. The CAE and VAE were trained both the non-vignetted and undistorted images $\mathbf{I}^u$. After training, the new test images representing aberrations $\mathbf{I}^a$ were passed to the autoencoders in order to reconstruct a new image $\mathbf{I}^a_r$. We have analyzed the mean squared error (MSE, eq. 11) between the reconstruction of aberrated $\mathbf{I}^a$ and $\mathbf{I}^a_r$ images.

$$\text{MSE} = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(\mathbf{I}^a(i,j) - \mathbf{I}^a_r(i,j) \right)^2 \tag{11}$$

According to our experiments, if MSE > 10, we interpreted that CAE/VAE detected the aberration. Otherwise, the aberration was not detected. The M and N stand for images dimensions (in pixels).

Lens Vignetting Detection. The results of the vignetting detection for the CAE and VAE autoencoders are presented in Tab. 1 and 2.

Table 1: CAE: Results of classifying the images that represent vignetting for all used cameras and lenses [%].

	vignetted	**non-vignetted**
vignetted	83.0	17.0
non-vignetted	10.0	90.0

Table 2: VAE: Results of classifying the images that represent vignetting for all used cameras and lenses [%].

	vignetted	**non-vignetted**
vignetted	89.0	11.0
non-vignetted	6.0	94.0

The average accuracy of the proposed autoencoders for classifying both vignetting and non-vignetting images is 89.0%. In the case of CAE, vignetted images are correctly identified in 83.0%, meaning that some images (17.0%) are not detected. The VAE performs even better, achieving 89.0% in recognizing vignetted images. Both models perform slightly better in recognizing non-vignetting images, correctly classifying 90.0% (CAE) and 94.0% (VAE) of them. However, there are still false positives, meaning that some non-vignetting images are incorrectly labeled as vignetting. Overall, both models show satisfactory performance but may require further improvements to better distinguish between difficult-to-classify cases.

Lens Distortion Detection. The results of the distortion detection for both autoencoders are presented in Tab. 3 and 4.

Table 3: CAE: Results of classifying the images that represent distortion for all used cameras and lenses [%].

	distorted	**non-distorted**
distorted	87.0	13.0
non-distorted	11.0	89.0

Table 4: VAE: Results of classifying the images that represent distortion for all used cameras and lenses [%].

	distorted	**non-distorted**
distorted	92.0	8.0
non-distorted	7.0	93.0

In the case of distortion recognition, the average accuracy of the proposed autoencoders is also equal to 90.25%. These results indicate that the proposed models offer reliable results of lens distortion detection. The test images representing the distortion were correctly identified in 87.0% and 92.0% of instances for CAE and VAE, respectively. However, 13.0% and 8.0% were incorrectly identified as non-distorted. On the other hand, 89.0% of non-distorted test images were correctly detected (CAE); 93.0% is the result of the VAE. Similarly, as in the case of vignetting, some images were incorrectly classified as distorted images.

4.4 Experiment II – Results of Digital Camera Identification

We have also tested, if the proposed autoencoders may realize the task of camera identification based on images. However contrary to Experiment I, the usage of CAE and VAE were changed. We have used only the encoder part of both autoencoders in order to generate the latent vectors. Then, the discriminator (introduced in subsec. 3.2) was trained with the latent vectors and made the final classification. In this experiment, all the tested methods were trained with the noise residuals calculated in the manner as shown in Eq. 10. Due to paper limitations, we skip presenting confusion matrices of cameras' classification results. The shortened results of classification are presented in Tab. 5.

Table 5: Results of classification.

Method	ACC [%]
CAE	91.0
VAE	89.0
Mandelli	92.0
Kirchner	92.0
Lukás	92.0

The results clearly indicate that all methods ensure very high identification accuracy. The proposed CAE achieves 91.0% identification accuracy, and the VAE 89.0%. In the other cases, the overall identification accuracy obtains at least 92.0%; the particular TPs are not lower than 90.0% for each camera. The Lukás et al.'s algorithm achieves almost the same results compared to CNN-based methods. This clearly confirms that all methods ensure reliable individual source camera identification. In the case of the discussed CNNs, the results are very similar to each other both in terms of identification accuracy and speed of learning.

All the methods require a similar number of training epochs to obtain the desired level of identification accuracy, which in this case was set to 2000. The learning rate was equal to 0.01, the Adam optimizer was used.

Speed of Learning. We have compared the time needed for training the proposed autoencoders and CNN-based methods. Results may be seen in Fig. 3.

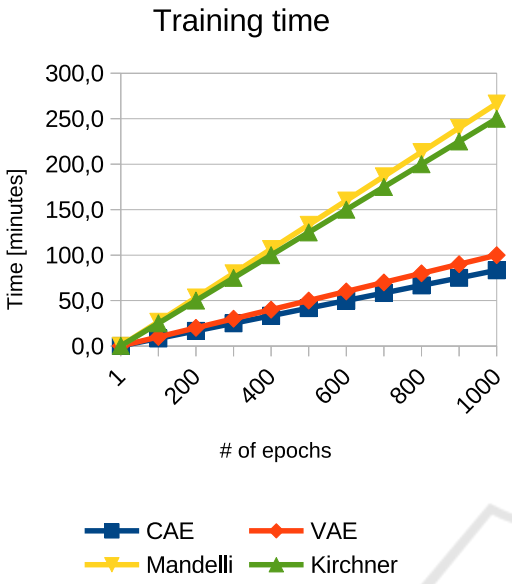

Figure 3: Comparison of time needed for learning 1000 epochs.

Results indicate that training the CAE and VAE requires less time per epoch than using state-of-the-art CNNs. One epoch using the proposed AEs is passed over 0.1 of a minute while using CNN turns to about 0.3 of a minute. Therefore, the overall time for passing 1000 epochs requires about 100 minutes for the proposed AEs and at least 250 minutes (more than 4 hours) for CNNs. Thus, it confirms the advantages over the literature methods.

5 STATISTICAL VERIFICATION – EXPERIMENT II

In this section, we analyze the TP values between the proposed autoencoders, Mandelli, Kirchner, and Lukás et al.'s algorithms. We determine whether there exist significant differences in this data. For this purpose, we analyze the MAE, MAPE, and RMSE error values, as well as the statistical verification of the defined hypotheses. The statistical verification concerns the results presented in Experiment II.

5.1 Error Analysis

We compare the TPs obtained by the proposed method and state-of-the-art methods by calculating estimators including mean absolute error (MAE, eq. 12), mean absolute percentage error (MAPE, eq. 13) and root mean square error (RMSE, eq. 14).

$$\mathrm{MAE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - x_t}{n}\right| \tag{12}$$

where x_t is the actual value, y_t is a predicted value and n is the number of observations.

$$\mathrm{MAPE} = 100\frac{1}{n}\sum_{t=1}^{n}\left|\frac{x_t - y_t}{x_t}\right| \tag{13}$$

where x_t is the actual value, y_t is a predicted value and n is the number of observations.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(x_t - y_t\right)^2} \tag{14}$$

where x_t is the actual value, y_t is a predicted value and n is the number of observations.

The results confirm that the classification using the proposed autoencoders achieves similar results as state-of-the-art algorithms. The MAE values obtain 1.05-1.12; the RMSE receives from 1.23 to 1.47, and the MAPE measure reaches from 1.15 to 1.20. Such small values mean that TP results obtained by the proposed autoencoders compared with other methods do not differ more than 1.20%. Also, none of the measures exceed 1.47, which we find satisfactory. The results of the analysis are shown in Tab. 6.

Table 6: The values of MAE, MAPE, and RMSE measures of the proposed CAE/VAE against state-of-the-art methods.

	MAE	**MAPE**	**RMSE**
Mandelli	1.0588	1.1483	1.3284
Kirchner	1.0588	1.1510	1.2367
Lukás	1.1176	1.2034	1.4753

5.2 Hypotheses Verification

We have checked if there exists a statistical difference between the results of classification by the CAE, VAE, and methods by Mandelli, Kirchner, and Lukás. All tests were performed at the significance level $\alpha = 0.05$. The first step is the normality test. The hypotheses are defined as follows:

- $\mathcal{H}_0$: Data represent the normal distribution;
- $\mathcal{H}_1$: Data does not represent the normal distribution.

We use the single-sample Shapiro-Wilk (SW) test. Results are presented in Tab. 7.

Table 7: Normality test results. The p-value less than the significance level leads to the rejection of the null hypothesis.

Data	p-value	test statistic S
CAE	0.001	0.771
VAE	0.002	0.792
Mandelli	0.004	0.817
Kirchner	0.004	0.817
Lukás	0.004	0.817

The critical values for a population consisting of 17 samples is $S_c = 0.892$. To assume compliance with a normal distribution test, the test statistic S should be greater than S_c. However, since $S < S_c$ for all considered data, we reject the null hypothesis about the normality of analyzed data. It is also confirmed by the p-values, which are much smaller than the considered significance level. Therefore, for further analysis, we use the Kruskal–Wallis ANOVA non-parametric test. The Kruskal-Wallis ANOVA test (also called the Kruskal-Wallis one-way analysis of variance for ranks) is an extension of the U Mann-Whitney test to more than two populations. This test is used to verify the hypothesis about the insignificance of differences between the medians of the studied variable in several ($k > 2$) populations (Kruskal and Wallis, 1952; Kruskal, 1952). Its hypotheses are defined as follows:

- $\mathcal{H}_0$: Medians $\theta_1 = \theta_2 = \ldots = \theta_n$ of the data are equal;
- $\mathcal{H}_1$: Not all medians θ_n (for $n = 1, 2, \ldots$) of the data are equal;

The test statistic is calculated as shown in Eq. 15.

$$H = \frac{12}{N(N+1)} \sum_{i=1}^{k} \Big(\frac{\big(\sum_{i=1}^{n_j} R_{ij}\big)^2}{n_j} \Big) - 3(N+1) \quad (15)$$

where:

- $N = \sum_{j=1}^{k} n_j$;
- n_j – population cardinality (for $j = 1, 2, \ldots, k$), corresponding to the TP values of the CAE, VAE, Mandelli, Kirchner, and Lukás;
- R_{ij} – ranks assigned to the variable value ($i = 1, 2, \ldots, n_j$, $j = 1, 2, \ldots, k$).

The statistic has a χ^2 distribution with $k-1$ degrees of freedom. First of all, let us calculate the sum of ranks. For $N = \sum_{j=1}^{k} n_j$, we have $k = 5$; $n_1 = n_2 = n_3 = n_4 - n_5 = 17$, therefore we obtain: $N = 5 \cdot 17 = 85$. The sum of ranks is presented in Tab. 8.

Table 8: Sum of ranks for the Kruskal–Wallis ANOVA.

Data	Sum of rangs
CAE	611.5
VAE	685.0
Mandelli	892.0
Kirchner	782.0
Lukás	684.5

Next, let us calculate the test statistic for the TP values obtained by the CAE, VAE, Mandelli, Kirchner, and Lukás:

$$H = \frac{12}{85(85+1)} \cdot \Big(\frac{611.5^2}{17} + \frac{685.0^2}{17} + \frac{892.0^2}{17} + \frac{782.0^2}{17} + \frac{684.5^2}{17} \Big) - 3(85+1) = 1.58$$

Thus, the statistical test value of ANOVA analysis for the CAE, VAE, Mandelli, Kirchner, and Lukás is equal to $H = 1.58$. The critical value of χ^2 distribution for 4 degrees of freedom is equal to $F_c = 9.49$. Because

$$F < F_c,$$

there is no reason to reject the null hypothesis about the equality of analyzed medians. Therefore, there is no statistical difference between the proposed CAE/VAE, Mandelli, Kirchner, and Lukás. This may be interpreted that all considered methods follow the same high identification accuracy.

Summary. The verification of the obtained results using both MAE, MAPE, and RMSE measures, and hypotheses verification confirmed that the proposed CAE and VAE make it possible to identify cameras based on images with similar accuracy to state-of-the-art methods. The MAE, MAPE, and RMSE measures represent small values, thus it may be interpreted that differences between each data are negligible. Also, the hypotheses verification, using Kruskal-Wallis ANOVA revealed that there is no statistical difference between the results of the classification of the proposed method and the literature's methods, so one may assume that the classification is at the same level.

6 CONCLUSION

In this paper, we have proposed a method both for lens aberrations detection, as well individual source camera identification based on images. The solution was based on convolutional and variational autoencoders. Extensive experimental evaluation conducted on a large number of modern imaging devices

confirmed the high lens aberrations identification accuracy. Moreover, the proposed autoencoders may be successfully used for digital camera identification. The experiments, enhanced with statistical analysis, confirmed the high identification accuracy compared with state-of-the-art methods. Additionally, experiments revealed that using proposed autoencoders may even shorten the processing time by up to half.

In future work, we consider an extended autoencoder model for increasing the accuracy of lens aberration detection. We are also interested in identifying different types of aberrations, including dispersion, coma, and astigmatism.

REFERENCES

(http://hugin.sourceforge.net/). Hugin photo stitcher.

Baar, T., van Houten, W., and Geradts, Z. J. M. H. (2012). Camera identification by grouping images from database, based on shared noise patterns. *CoRR*, abs/1207.2641.

Bernacki., J. and Scherer., R. (2023). Imagine dataset: Digital camera identification image benchmarking dataset. In *Proceedings of the 20th International Conference on Security and Cryptography - SECRYPT*, pages 799–804. INSTICC, SciTePress.

Bondi, L., Baroffio, L., Guera, D., Bestagini, P., Delp, E. J., and Tubaro, S. (2017). First steps toward camera model identification with convolutional neural networks. *IEEE Signal Process. Lett.*, 24(3):259–263.

Claus, D. and Fitzgibbon, A. W. (2005). A rational function lens distortion model for general cameras. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 213–219. IEEE.

De Silva, V., Chesnokov, V., and Larkin, D. (2016). A novel adaptive shading correction algorithm for camera systems. *Electronic Imaging*, 28:1–5.

Ding, X., Chen, Y., Tang, Z., and Huang, Y. (2019). Camera identification based on domain knowledge-driven deep multi-task learning. *IEEE Access*, 7:25878–25890.

Goljan, M. and Fridrich, J. (2014). Estimation of lens distortion correction from single images. In *Media Watermarking, Security, and Forensics 2014*, volume 9028, pages 234–246. SPIE.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. (2014). Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Jiang, X., Wei, S., Zhao, R., Zhao, Y., and Wu, X. (2016). Camera fingerprint: A new perspective for identifying user's identity. *CoRR*, abs/1610.07728.

Julliand, T., Nozick, V., and Talbot, H. (2016). *Image Noise and Digital Image Forensics*, pages 3–17. Springer International Publishing, Cham.

Kingma, D. P. and Ba, J. L. (2015). Adam: A method for stochastic gradient descent. In *ICLR: international conference on learning representations*, pages 1–15. ICLR US.

Kirchner, M. and Johnson, C. (2020). SPN-CNN: boosting sensor-based source camera attribution with deep learning. *CoRR*, abs/2002.02927.

Kordecki, A., Bal, A., and Palus, H. (2017). Local polynomial model: A new approach to vignetting correction. In *Ninth International Conference on Machine Vision (ICMV 2016)*, volume 10341, pages 463–467. SPIE.

Kordecki, A., Palus, H., and Bal, A. (2015). Fast vignetting reduction method for digital still camera. In *2015 20th International Conference on Methods and Models in Automation and Robotics (MMAR)*, pages 1145–1150.

Kruskal, W. and Wallis, W. (1952). Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, pages 583–621.

Kruskal, W. H. (1952). A Nonparametric test for the Several Sample Problem. *The Annals of Mathematical Statistics*, 23(4):525 – 540.

Lanh, T. V., Chong, K. S., Emmanuel, S., and Kankanhalli, M. S. (2007). A survey on digital camera image forensic methods. In *2007 IEEE International Conference on Multimedia and Expo*, pages 16–19.

Lee, S. Y., Cho, H. J., and Lee, H. J. (2017). Method for vignetting correction of image and apparatus therefor. US Patent 9,740,958.

Li, R., Li, C., and Guan, Y. (2018). Inference of a compact representation of sensor fingerprint for source camera identification. *Pattern Recognition*, 74:556–567.

Lopez-Fuentes, L., Oliver, G., and Massanet, S. (2015). Revisiting image vignetting correction by constrained minimization of log-intensity entropy. In *Advances in Computational Intelligence: 13th International Work-Conference on Artificial Neural Networks, IWANN 2015, Palma de Mallorca, Spain, June 10-12, 2015. Proceedings, Part II 13*, pages 450–463. Springer.

Lukás, J., Fridrich, J. J., and Goljan, M. (2006). Digital camera identification from sensor pattern noise. *IEEE Trans. Information Forensics and Security*, 1(2):205–214.

Mandelli, S., Cozzolino, D., Bestagini, P., Verdoliva, L., and Tubaro, S. (2020). Cnn-based fast source device identification. *IEEE Signal Process. Lett.*, 27:1285–1289.

Marra, F., Gragnaniello, D., and Verdoliva, L. (2018). On the vulnerability of deep learning to adversarial at-

tacks for camera model identification. *Sig. Proc.: Image Comm.*, 65:240–248.

Moosavi-Dezfooli, S., Fawzi, A., and Frossard, P. (2016). Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2574–2582.

Papernot, N., McDaniel, P. D., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. (2016). The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*, pages 372–387.

Park, S.-W. and Hong, K.-S. (2001). Practical ways to calculate camera lens distortion for real-time camera calibration. *Pattern Recognition*, 34(6):1199–1206.

Ray, S. (2002). *Applied photographic optics*. Routledge.

Taspinar, S., Mohanty, M., and Memon, N. D. (2016). PRNU based source attribution with a collection of seam-carved images. In *2016 IEEE International Conference on Image Processing, ICIP 2016, Phoenix, AZ, USA, September 25-28, 2016*, pages 156–160.

Tiwari, M. and Gupta, B. (2018). Image features dependant correlation-weighting function for efficient prnu based source camera identification. *Forensic Science International*, 285:111 – 120.

Tuama, A., Comby, F., and Chaumont, M. (2016). Camera model identification with the use of deep convolutional neural networks. In *IEEE International Workshop on Information Forensics and Security, WIFS 2016, Abu Dhabi, United Arab Emirates, December 4-7, 2016*, pages 1–6. IEEE.

Yao, H., Qiao, T., Xu, M., and Zheng, N. (2018). Robust multi-classifier for camera model identification based on convolution neural network. *IEEE Access*, 6:24973–24982.